**COS340 Programming in Python**

**Course Project**

Lyuboslav Gigov

28.04.2024

**Project Description:** The project is designed as an interactive book recommendation system for bookstores, aiming to assist customers in selecting books that best match their individual preferences and criteria. Users can specify various parameters such as age, maximum price, genre interests, and other preferences. The system evaluates a collection of books, calculating a suitability grade for each based on the user's input. It dynamically adjusts the selection process according to user feedback. The goal is to make the book selection process more efficient and personalized.

**Class Structure:** The project is built around the central **Book** class, which encapsulates common attributes such as title, author, price, publication year, and ISBN. This class also defines methods for setting and retrieving these attributes and includes a method for calculating the book's suitability grade.

Derived from the **Book** class are **Fiction** and **NonFiction** subclasses, which represent specific genres of books. The **Fiction** class branches further into **ScienceFiction** and **Romance** classes, each with specialized attributes and behaviors that reflect their respective genres. For example, the **ScienceFiction** class includes factors like scientific accuracy or action level, while the **Romance** class considers emotional depth and realism.

The **NonFiction** subclass encompasses **Biographies** and **Encyclopedias**. The **Biography** class tailors its grade calculation to the relevance to user interests. The **Encyclopedia** class is tailored for users seeking factual and educational content, adapting its recommendations based on the required educational level.

All classes use polymorphism to tailor the grade calculation process according to genre-specific criteria, ensuring that the book recommendations are as relevant as possible to the user's stated preferences. The **Bookstore** class manages the bookstore’s inventory, while utility classes like **BookCriteria** capture user preferences, forming the basis for the recommendation logic.

This is the three-level hierarchy of the project:

A diagram of a book

Description automatically generated

**Special Functions and Algorithms:**

* **Fuzzy String Matching:** Used in the **match\_keywords** function to compare user input with a pre-defined list of keywords, accounting for close matches and misspellings to better identify user interests.
* **Polymorphism:** The **calculate\_grade** method within the **Book** class and its subclasses is polymorphically designed, allowing each subclass to override this method with genre-specific grading logic.
* **Regular Expressions (Regex):** Applied in the ISBN setter to validate the ISBN format, ensuring consistency and accuracy in the book data.
* **Sorting Algorithm:** Books are sorted based on their grades, using Python's built-in sorting functionality, which is made possible by the overloaded **\_\_lt\_\_** method in the **Book** class to compare books by their grades.

**Instructions for Working with the Program:**

1. **Initialization:** Launch the program by running the **main.py** script. This is the entry point of the application and will start the interaction process.
2. **Entering Preferences:** Follow the prompts to enter your age, maximum price for a book, gender, and other interests or keywords related to the type of books you are looking for.
3. **Receiving Recommendations:** Once the information is entered, the system will process the bookstore's inventory and output the top five book recommendations that match your preferences.
4. **User Feedback Loop:** If the system prompts for confirmation regarding the recommendations or additional information, make sure to answer correctly and honestly to further refine the results.
5. **View Results:** After processing, the program will display the most suitable books in a tabular format. You can restart the program to enter different preferences and receive new recommendations.

Should any errors arise, please turn to the bookstore personnel to assist you. Make sure that:

* The required files with keywords are placed in the program’s directory
* The .txt file containing the bookstore inventory is properly formatted
* Your input does not contain major spelling mistakes (the program is designed to work despite such errors, but nothing is perfect!)

**Important Note:** I have used the following python libraries: **fuzzywuzzy**, **flake8**, **tabulate**, **unittest**, **re**, and **sys.** For the **docstrings**, I have used the **Google** format.